**CRAIG:** Hi, I'm Craig Smith and this is Eye on AI. This week we're talking about machine learning operations called ML Ops for short, the plumbing that allows data scientists to build machine learning models. This podcast is sponsored by an MLOps company, ClearML, and so what better way to explore this burgeoning field than speaking to our sponsor? My guest, Moses Guttmann, founded ClearML, and so I asked him to describe the evolution of the MLOps industry over the past few years and ClearML's contribution to it.

I hope you find the conversation as informative as I did.

**CRAIG:** What I generally do is have the guests introduce themselves, give their educational background, how they got into machine learning. Maybe you can give the evolution of MLOps and what ClearML does and the market and that sort of thing. I've spoken to a number of people about MLOps. I had Andrew Ng on the podcast and Landing. AI has become basically an MLOps platform.

**CRAIG:** And then a guy named Bratin Saha who runs the SageMaker platforms at Amazon. I'm curious personally, as well as for the podcast how that market looks from a founder's point of view. Why don't we start Moses with you introducing yourself?

**CRAIG:** Give us a little bit background and then we'll talk from there.

**MOSES:** Sure. So, I'm Moses, I'm the CEO and founder of ClearML. Before ClearML, I had another company doing computer vision, 2D to 3D semi-automatic conversion, which actually relates to what we see now in deep learning. So, a lot of very similar tool sets, design, and workflows. Before that I was pursuing my PhD.

**MOSES:** So, we started a company inside the university, and before that I served in a cyber intelligence unit.

**CRAIG:** For the IDF?

**MOSES:** Yes. Correct.

**CRAIG:** Oh, that's interesting. So, you were pursuing a PhD at Tel Aviv university, specifically in computer vision?

**MOSES:** Yeah, I did my master's, published a few papers and we started a company actually based on one of the patents that we wrote as one of the papers.

**MOSES:** And then from that I moved to the industry. And left the academic world.

**CRAIG:** Yeah. And the first start-up?

**MOSES:** So, the first one was Qptical, which is a wordplay on optical and cues, which you can think of as annotating data sets, which is basically the same thing.

**MOSES:** We can take an image and add depth to it by learning how a couple of other images with depth look like. So, we can take a 2D version of video clips and data, et cetera, and make it 3D.

**CRAIG:** Is that separate from the data annotation?

**MOSES:** Well, it kind of predated data annotation. So actually, we started with classical computer vision. Let's take some algorithms and build a data layer on top of them. And we want that data layer to have a human in the loop.

**MOSES:** So, they add their capabilities of separating different objects and laying out the objects in space. And then we want to get an algorithm to build the 3D space around it, et cetera, then project it to different angles, but that entire product actually meant building a lot of infrastructure.

**MOSES:** So, the algorithmic part of it was relatively small, and this is exactly what we're seeing today with any machine learning product. Basically, the product itself, from a product perspective, machine learning is just one part of that product. And it needs a lot of infrastructure to make sure that part actually works, but it is only one aspect of that entire final product for the users. And since that deep learning algorithm is very complicated, you need to have a lot of infrastructure to make sure that it actually works.

**MOSES:** Back then, it was not machine learning operations it was basically just software infrastructure but these days we would call that MLOps.

**CRAIG:** And then because you were building this infrastructure, you decided to productize the platform? How did you then segue into MLOps full time?

**MOSES:** So that company actually got sold together with that product. That was 2016 or so, and deep learning exploded. And that's when I saw, we have new types of algorithms, and the infrastructure is even more important.

**CRAIG:** And you're part of a movement in MLOps in, machine learning operations.

**CRAIG:** Can you talk about how that developed, not only for you guys, but in the market generally. And

**CRAIG:** who the other players are.

**MOSES:** So, in the beginning, let's say 2019 or so, it was only a few players. A lot of them are actually presenting a new development paradigm that is shifting from the classic software development paradigm. So, the classic software development paradigm is very linear.

**MOSES:** You start with someone defining a feature or a bug, and then you fix it or build a new feature. You move it into QA, someone tests it, et cetera. You fix the code. It passes QA, and then you deploy it. Great. You have the new version of the product. But you'll never go back. Once you write a feature, it is there, and it is working.

**MOSES:** There is no reason why it will stop working. When we think about machine learning and deep learning, those features are constantly evolving or constantly breaking, depending if you're more optimistic or pessimistic, but that's essentially the same and you always have to babysit those. You are not just throwing them out there and saying, okay, now we move to the next one.

**MOSES:** Because things constantly change, and the products constantly change, and the data constantly change. And those changes introduce something into the model and causes it to degrade or it is not stable as traditional software. So traditional software is very stable once it's working, and in order to solve for that different paradigm, where basically you're just deploying things that are not stable enough and you have to babysit from the get-go, and you're just waiting until they explode.

**MOSES:** That's what we're doing. It means that you have to understand that you're coming back to development constantly. Which means you cannot think of it as a linear project management problem. This is a cyclic problem where you're constantly developing and you have to create visibility and production to the people who actually wrote the initial code, which is also problematic. Usually visibility into production, this is forensics. If something crashed, I want to go back and see why. But now we have telemetry that actually tells us something about the internals. Just looking at it.

**MOSES:** If this is not your model, you have no idea what it means. Is it good that the score is up or down? Should it be that volatile? No one will know that, only the people who wrote that initial code base.

**MOSES:** And that's when we realized as an industry that we have to start to build infrastructure for that. And I think that the first step was understanding that the development part is actually changing, and it’s longer enough to store your data. You need to store your metrics alongside that data.

**MOSES:** And that's when I think it was ML flow. And then we were the second open-source offering and probably the only ones there until today lasting that long. And then a few other paid offerings as well because it was the lowest hanging fruit. It's just the first step towards this entire process.

**MOSES:** And that's where it started.

**CRAIG:** A very interesting analysis because I haven't thought of it that way. I was initially introduced to MLOps through a data labelling platform called label box. And those guys were in industry, planet labs, where they handle a lot of imagery. And they had to build from scratch a labelling platform and workflow for their imagery. And these guys realize that if we're having to build that from scratch everybody, who's building a computer vision product is having to build it from scratch. There should be an off the shelf solution that everybody can use.

**CRAIG:** And so, I've thought of MLOps as building the tools that otherwise data scientists are going to have to build themselves and repeat across projects or companies. But it sounds to me that you're talking more holistically about monitoring models once they're deployed and being able to go back and adjust the data as there's drift in the model.

**CRAIG:** Everyone. I talk to in MLOps says, ' oh, we have an end-to-end platform.' But once I start talking to them in detail, it seems like they deal with the labelling or the data prep or the neural architecture search or, deployment or monitoring.

**CRAIG:** So, did people start in this industry taking bites at different parts of that pipeline and how did you guys develop.

**MOSES:** Two things that you mentioned that are very interesting. One, you said it sounds holistic, you hit the nail on the head with that one.

**MOSES:** That's actually how we look at things. And the second is the observation that everyone is saying that they're end to end, but they are kind of a point solution approaching the problem from a very specific angle. And sometimes they have other point solutions in their set of products, but that does not mean that they are fully integrated, meaning holistically, they work together.

**MOSES:** So, you think about the cloud providers, they're very good at point solutions, but then it's on the user to actually see everything together. We took a different approach and that's based on our experience, that the only way that you can actually make it work is if you have the underlying infrastructure unified, if you will – basically everyone is using that same underlying infrastructure so you can pass data information from the different stages very quickly. Otherwise, every time you move from one step to another, data labelling to training the model, to testing it on new data, and then feeding back to the data labelling, you have to basically integrate with different steps, which means a lot of boilerplate code for users.

**MOSES:** And every user will have to end up with a different best-of-breed solution that they came up with and that means that at the end of the day, you have a lot of code to maintain that does basically very little; it stitches different modules together. Then you end up debugging the stitches. Then you're not spending time on the things that would actually make your product better from a company’s perspective. And that's a very different approach that we took. We said, we will just build the infrastructure, like really the basic stuff. And then we will build on top of it with a few ideas on entities and how they work and how you pass data. And we'll just add the different modules based on what users are struggling with, but they all sit on the same infrastructure. So, it is very easy to add those different modules. You don't actually have to reinvent the wheel every time. You have the entire tool set available for you, which means building and new modules becomes very quick. You can very easily add them and that actually means that from a user's perspective, it is very easy to go back and forth between the two different stages or three or four or five, depending on your workflows. So, you can always go back, and you can always jump.

**MOSES:** You never actually have to stitch your code into a different stage just because you want to jump from one to another. And this saves a lot of time in the development of models, because now you can actually test them very quickly because at the end, we always fail the first time we try. Probably the second and third, but if we do that kind of iteration quickly enough, then we can actually get them to be deployed as part of the product.

**MOSES:** Because only then we will say we have an idea on how to actually improve it. So that’s the next step of the evolution of your models.

**CRAIG:** And what module did you guys start with?

**MOSES:** Oh, like everyone – experiment management, as that was the lowest hanging fruit. It was very obvious that users are looking for a solution, as back then the standard was TensorFlow.

**MOSES:** And say a year before that, say 2018 or 17, people thought, oh yeah, I have TensorBoard that's enough. But over time they realized it's not a very good way to store your data. It's very local and explodes very quickly. So, it's huge in terms of size.

**MOSES:** It's very easy to mess everything around or the opposite of that – actually create order and manage the entire thing. So, after two years of people using it, everyone realized it's nice to have, but it's not a very good solution that scales, it definitely cannot store it long term. As a long-term solution, it’s not very good. And we thought, okay, that's the first module that we will open source. We designed this entire thing in a way that is very modular, again, a joint infrastructure.

**MOSES:** And we said, okay, we're publishing the entire thing. So, the client side, the backend side, the web interface, which was back then was the exception. So even the web UIs: open source. I cannot think of a lot of other projects doing open-source UI. So, everything became open source because it was important for us first to start to create the community.

**MOSES:** But also, because that's just the first step among many to build this entire flow.

**CRAIG:** And then the modules as they're being added, how much of that is coming from Clear ML engineers, and how much is coming from the open-source community?

**MOSES:** So, features are built based on the open-source community feedback.

**MOSES:** People ask, and we brainstorm ideas, and we say, hey, we thought about adding this or that. And what do you guys think? So, there are always discussions about that, but the bulk of the code base is still coming from ClearML even though we have a lot of people contributing code. It is a very complicated system with joint infrastructure that actually spans different repositories, doing a lot of back and forth in the web interface and the backend and the Pythonic clients and the orchestrators.

**CRAIG:** the market, you were saying about 2019 or so this MLOps market started to grow and there are so many MLOps companies now, how do you see the market? And within that, as I said, there's some that, that deal with one particular.

**CRAIG:** End of the pipeline or one section of the pipeline. There are others that are more end to end. I just had Bratin Saha from Amazon talking about SageMaker. How do you see the market and how does ClearML fit into that? And how do you compete with somebody like Amazon?

**MOSES:** Right. I think that you mentioned before. And I think again, that was a very accurate observation: everyone is saying that they are end to end. I think that's in the MLOps space. You have two options to look at the MLOps space or at least that's how we look at it. So one is, these are just developer tools.

**MOSES:** Like any other developer tools, you have a lot of selection, and some are open source, some aren't. And then the other is for a large company, they want full ownership of the entire end to end, which means they want a one-stop shop, which means you have to say that you are end to end.

**MOSES:** And I think this is why a lot of solutions are saying that they are end to end, because it is easier to say, oh, just come here – we'll cover everything that you need. I think that the reality is a bit more complicated as you mentioned, like it might say end to end, but it started with a very specific angle and all the rest of the modules are very close to that specific angle.

**MOSES:** And I think that for a lot of them, it's a collection of solutions, rather than a unified one. So if, you mentioned Amazon SageMaker, it has a collection of tools developed inside Amazon sitting on top of the Amazon infrastructure. But these are a set of tools that are available for machine learning practitioners.

**MOSES:** They are not a unified holistic solution. You can use one and then the other, and then basically you can replace one with a different one, at least in theory, but they are not holistic in the sense that they are all transparent: You start here. You end here. You don't have to worry about it. We'll take care of you.

**MOSES:** Rather, it's always, oh, you start here and then you build a new configuration file, and then you build a new container, and then you move to the second stage and then you create a new container, and your artifacts are in the S3. And then, oh yeah, we forgot to tell you, you have to write it down. And then everything is, yeah, we cover everything, but as the user, you have to do a lot of the heavy lifting in order to actually stitch it into something that actually works.

**CRAIG:** And ClearML’s approach them?

**MOSES:** So ClearML’s approach is basically you have a control plane, which is the back end, which is being used in a lot of very different ways. But you can always access it.

**MOSES:** Everything is there. And you never have to leave the ecosystem, but you can always leave the ecosystem. So, the way that it works is once you integrate your code, which basically means that you just add two lines of code into your Python logic, everything that you do is logged by the system so you can always trace it and then you can decide what you want to do with it.

**MOSES:** So, you can have your model over here, but you don't have to, you want to move it somewhere else? That is all yours. Here's an API. You want to move it to a different storage solution, just configure it to work differently. That's the default. You don't have to, but you can do the orchestration with ClearML.

**MOSES:** Here, you click here. We'll spin an instance, and this is for you. We'll spend your code. You don't want that. You can containerize it. Here is a utility to do that. Or here is how we integrate. Something else you want? Airflow? here you go. It's a login into ClearML. So, you can always trace it back and you can always see the entire flow, even if you're using different tool chains and this environment.

**MOSES:** So, on the one hand, we don't limit you. On the other hand, you still have a unified view of all the different things that are happening inside your cycle.

**CRAIG:** Can you walk me through the. The ClearML platform when someone comes to it and I believe you have a ??Freemium model, you can register and start using it.

**CRAIG:** What's the first step in building a model clear through to a monitoring the model and then the iterative loop when things change.

**MOSES:** Sure. We have the open-source version, which basically means you spin everything yourself. This is one option. This is very important. So, companies can spend their entire service internally, no leakage of data.

**MOSES:** You don't have to worry about security. It's fully secure. These days that’s very important for privacy, data sharing, et cetera. I think this is key. The other option is you go, and you register for our free tier, which basically is just one click – you register the Google account, GitHub, or whatnot to get an access to the backend.

**MOSES:** So, the free tier is basically, we just do the backend for you, compute you still provide for yourself, and we can manage it for you, but we don't do the compute for you. We might have a joint offering with a hundred percent green cloud provider very soon, but still, this is your compute. That's basically the bottom line.

**MOSES:** We are providing you a control plane over whatever you're doing. You sign up and then the first step is we're assuming you have a piece of code that you already worked on. So, our assumption is you already have something that you're working with your way of developing things we don't want to interrupt with whatever is working for you.

**MOSES:** We want to integrate with that. So, if this is a Jupyter notebook or a via code or PyCharm, or your Git repository, it doesn't matter, you go into your code base and you add two lines of code in order to integrate with ClearML. So, one thing that’s important is the package itself. And the second one is just initializing it as part of the first step of your execution of the process.

**MOSES:** In terms of training a model or testing it, you just call the initialization function from that point onwards, everything that you do inside your process on your machine is being monitored and sends to the control plane. So, the first step is obviously you have full visibility. I know what I did. It stores everything for me.

**MOSES:** So, it links to my Git repository, or it stores my Jupyter notebook. t stores the Python packages that I'm using inside my code base. So, I later understand, oh, I was using that specific version, that specific package stores, my arguments that I pass configuration files, everything. The ideas that the first step is, let's make sure that when we are in the development stage, we can always roll back and understand what we did a week before to better understand where we're

going with our model and what's going on. And the second or a derivative of that, we can now share it with our colleagues. Hey, I'm working on this one. What do you think? It's basically, hey, here's a link to what I'm working on. Now you can see it. So, think of it as sharing with colleagues, but also in terms of managing a team, right?

**MOSES:** When you manage a team, you have zero visibility looking at a ticket system for features. If you think of software development, it’s not the same as model development, you can always say, oh yeah, it's work in progress. Which tells you nothing about when it'll be done. You want to create that visibility. That is part of the development process.

**MOSES:** For example, here are a few graphs from last week. That's where we're at. We're stuck at a loop or whatever. So that's the second step of this visibility. And then once you have that, then you can move to the more complicated step, which is CI/CD for model training, right? We want to introduce the automation.

**MOSES:** Without automation, there is no way you'll end up with a model that actually works at scale. Like it might work one time. we got lucky, we were at the office, it works, but what will happen the second time that something changes? If we reproduce it, can we automate this fast? So, the second step of automation is important. And people think about that as, oh, that's not our problem.

**MOSES:** We'll just call someone else a software development person. And then that software engineer will come up and say, oh, I'll write you a Jenkins script. And I'll automate this entire process and I'll reinvent the wheel and I'll create a container. And, but then what will happen if you change a code, you call them again and they will not come by the third time.

**MOSES:** They'll not answer your calls. And the idea is that, how do we take that process of moving code into something that we can automate very quickly? Because this is the key to success here. And this is the second module of ClearML. So, we call it ClearML Orchestrate or ClearML agent. It's basically saying, okay, we have all the information that we need from the first step.

**MOSES:** Now let's build something that will basically replicate the environment, make sure that you have the right code, the environment, the packages, and the parameters, and the arguments. I'll touch on that in a second. And how do we run it on a remote machine? Once we have that, basically we can automate the entire thing, which means now I have many machines.

**MOSES:** I can just send jobs to get progress reports the same way I did manually. Now I can do it remotely because it's essentially sending the metadata into the control plane, which I always have access to. And now I can control this flow. And the only thing that was missing was changing parameters. But again, the way that ClearML works is it tracks everything alongside your arguments and configurations.

**MOSES:** It gives you the way to edit it and change it. So now you have a full unified interface to start developing and automating things. And the automation is where you actually get the value from

**CRAIG:** And then, once it's deployed, there's monitoring.

**MOSES:** So, the automation part needs to end up with something that you can actually use.

**MOSES:** So as part of the ClearML magic, everything that you store locally, we add as an artifact of the model. Again, you don't have to worry about the model repository and making sure that you can trace back artifacts and then obviously use them, right? Because when I store something locally, if I want someone else to use it, I have to upload it.

**MOSES:** And I have to remember where it was uploaded to, et cetera. This happens automatically for you. So, in the second step of automation, what we do is we create the model repository for you, right? Every time you store a model, we upload it for storing in the cloud, linking it to the actual code base that created it, and giving it an entity name.

**MOSES:** So now you can use it and actually download it and use it for deployment. And when you think about deployment again, you can either say, okay, I have my own deployment. I know how to build my deployment solution. Here's a Rest API for you or Python API. Now you can query all the different models by tags or status.

**MOSES:** Get a link down to the model, deploy it yourself. The other option is you use ClearML to do that, which essentially does that automatically for you. Connects to the same control plane and says, okay, here's your model. Do you want to attach some codes to that model? Because this is important, but the reason people want to deploy their own models with their own infrastructure is because they realize in order to deploy the model, I probably need to do something before and after the model is running.

**MOSES:** So, if the model gets a link to an image and you want to classify the image, someone needs to download that image, right? No one will just provide you the bytes of the image. If this is a text and someone needs to tokenize a text, you want to separate it into different words. So, pre-processing should come with a model itself, whether it's not part of the model, this is just Python code that needs to be running before or after.

**MOSES:** So, when we designed ClearML we said, okay, let's make sure that we always couple them together, but allow you to control it. Usually, it's a very small piece of code. And you want that to always go hand in hand with your model, but also to be able to deploy it easily

**MOSES:** Now you can actually deploy the model while it is running. So, the idea was first to get that. And then the second is add monitoring out of the box, basically. We don't want you to have to send to us the type of data that you want us to monitor. Basically, you provided everything because there is an in and there is an out, and we have access to those.

**MOSES:** Again, same approach of “here is some magic that basically traces back everything. Just tell us what you want us to do with the data, it's already there.” And we'll just make sure that we push it into large databases with dashboarding on top for you to actually build your own monitor on top. We'll do the heavy lifting for you, and this is exactly how it works

**CRAIG:** on the other end, preparing the data and labelling the data.

**CRAIG:** Is that done within the ClearML ecosystem as well? Or is it assumed that you have your data prepared?

**MOSES:** Right, this is a very good question. When we started, that was part of our internal first modules that we built, because we had to, as you mentioned, back then people had to build their own. These days, it depends on the actual type of data that you have.

**MOSES:** So ClearML has what we call ClearML data Ops, so ClearML data that will store and version your data at scale. So, think of it as differentiatable storage that only links to that exact control plane. So, you can actually query it and fetch it, etcetera. People tend to think about it as Git for data. I am not a fan, even though it creates good intuition, but it is not the same. It's close, but it's not the same, but versioning is there.

**MOSES:** We're just saying the approach is not the same as handwritten code, right? You create versions or actual snapshots of the data in very specific use cases, you actually merge or fork, so different use cases, but the same concept. So, we support that.

**MOSES:** So, we store that for you. and we support a very specific type of visual data annotations and basically any unstructured data annotations. So, when we thought about annotation, when we first designed this entire system and again, built into this entire control plane, we said, okay, we'll have a very specific type of metadata that we want to store on top of the raw data, which makes a ton of sense if your raw data is unstructured.

**MOSES:** So unstructured, you think of it as a link to an image or an auto follow, or even a text file. We will not index the content of that, right. Some things cannot be indexed like an image, but the metadata on top, we want to index that. And what we did is we created an entire differentiatable database storage on top of a database.

**MOSES:** So, we can actually abstract the data for you. And as part of it, we actually allowed you to visualize the data. And then we can already see the labels and the boxes. So, we'll allow you to edit those as well. But it actually started from, we just want to understand the metadata and how to manage it rather than this is a tool to manage large scale annotation groups, doing annotation for something very specific.

**MOSES:** It became a tool to debug annotation with an interactive interface and the ability to actually explore the dataset is part of this entire pipe.

**CRAIG:** Is there anything that ClearML does not do that you're looking to add?

**MOSES:** So, we are now adding reports, which is a better way to basically write something for the executive level or present something internally. Things that were brought again by the community. Everyone wants that, so we're adding that. And we're also adding that capability to integrate with, again, external tools. This is very important for us as an approach. We don't want to create that closed garden approach.

**MOSES:** We're totally against that. So that will be integrated with any other external editing tools. So, you can just embed it to whatever tool you want to build or inside the system itself. What we are not building is, for example, a feature store. So, if you think about structured data and tables and their requirements for a feature store, most probably like an online feature store, actually querying and building the SQL query.

**MOSES:** This is actually not us. We integrate with other open-source solutions for that. And we have a lot of people from the community actually building the integration and saying that was very easy to build. But this is not part of the ClearML holistic approach.

**CRAIG:** How big is the community?

**MOSES:** I can say that we have thousands of servers, open-source servers, up and running.

**MOSES:** We have thousands of stores, a very large Slack community, extremely active. So, on a daily basis, probably hundreds of messages and requests and people talking. So, it's a very active community.

**CRAIG:** How many other open source MLOps solutions are there. And how does your community stack up with other communities and are those communities clustered around different kinds of models? Maybe computer vision or natural language processing, or are they regional?

**CRAIG:** Is there like a big community in Europe and the Middle East, maybe another community in Asia. If you're an MLOps engineer, how do you discover ClearML? How do you decide to join ClearML as opposed to some other

**MOSES:** I think your point on the different type of problems is probably correct.

**MOSES:** So, NLP versus computer vision versus structured data. So, you do have communities being created around solutions in these specific areas. Usually, they work together with other open-source solutions to build a solution on top.

**MOSES:** So, a workflow on top of the different tool sets. I cannot think of either open-source tools that are that extensive in terms of the things that they cover. Of course, there are other solutions based on the specific problem. You can always find multiple solutions and some of them actually have active communities.

**MOSES:** Usually they're very, narrow to solve a very specific problem. And sometimes it's attached to the type of data that they're working on. And I guess that's the best way to, if I wanted to think about the different clusters, I would say first is the type of data or the type of problem that you were working on.

**MOSES:** And only later kind the type of product you're working on, the kind of the derivative of that data. Such as art versus robotics. If you think about computer vision or kind of our artistic approach versus our media or content versus robotics, as I need to handle myself in the physical world and the same goes for NLP and structured data.

**MOSES:** So different verticals kind of fork from one another into different sets of packages and communities.

**CRAIG:** is your community focused on computer vision because that's where you guys came from.

**MOSES:**

**MOSES:** It's actually quite agnostic. It is very GPU-oriented A lot of the value from automation, scale, and complexity comes from the fact that, if you are running GPU or deep learning, training jobs, you actually have to do a remote execution at a very early stage of the development because it doesn't scale otherwise.

**MOSES:** In the development stages only when you think about, okay, we should probably have some QA process or QC process for the model, or we should have some provenance on the code base that actually created it, or when you actually think about, okay, the next stage is going to actually be deployment. Are we using that model itself?

**MOSES:** So, at a late stage, classic machine learning structure data, or from early stage, anything that is deep learning.

**CRAIG:** And then how does ClearML you're open source, but how do you guys make your money? Presumably then you have a premium offering that either has additional features or you give support.

**CRAIG:** Do you have a consulting end to the business?

**MOSES:** So, we are not a services company. We have the open source. We have the free tier. We have the pro tier, which basically is totally transparent self-service. And then we have the scale and enterprise tiers, which are basically, will you pay for your yearly license?

**MOSES:** We provide support. This is not a service just support. And we manage the entire thing. So, things like VPC on-prem deployments, a lot of more sophisticated features. Like SSO and better Kubernetes integration, et cetera that are just not part of the open source.

**CRAIG:** The community then is growing through word of mouth, or I don't know how open-source ecosystems grow.

**MOSES:** Word of mouth basically. That's the only way to grow the open-source community. So basically, friends bringing friends and telling other people, et cetera, it's very organic.

**MOSES:** Marketing always helps. Unfortunately, a lot of the very large companies that are using ClearML to build some really awesome stuff will not say what their internal development relies on for, a good reason, but you can go check out our website.

**MOSES:** There are some nice logos there.

**CRAIG:** And what do you see for the future of both ClearML and MLOps? I'm also very interested in the no code movement and, automated code generation, everything seems to be trending toward increasing automation, drag and drop interfaces and things like that.

**MOSES:** I think that you're correct. It progresses to where it's commoditizing machine learning. I think that as time goes by, it'll be more of another field of software engineering. Obviously, if you go down into the details, then this will not change, but you'll get a very broad set of users that are just software engineers, because it is that easy to use. You don't actually have to understand the mechanisms of how things work. These days in a lot of cases, machine learning is added as an afterthought, instead of built into the product. That's the trajectory it's taking, because we are seeing the commoditizing actually happening with the ability of the frameworks to add another layer of abstraction, another layer of simplification, which just ends up as another software module for software engineers to use.

**CRAIG:** That's it for this episode. I want to thank Moses for his time. If you want to learn more about ClearML’s suite of machine learning tools for AI developers, check them out at clear.ml. As always, you can find a transcript of our conversation today on our website, eye-on.ai.

And finally, remember, the Singularity may not be near. But AI is about to change your world, so pay attention.